

1 Вычислительный эксперимент

Был проведен вычислительный эксперимент на кластере «Cyberia»¹ Межрегионального вычислительного центра Томского Государственного Университета. Данный кластер состоит из 282 узлов, на каждом узле находится два двухъядерных процессора Intel Xeon 5150, 2,66ГГц.

1.1 Тестовая библиотека

Для вычислительного эксперимента была использована библиотека A Database of Graphs for Isomorphism and Sub-Graph Isomorphism Benchmarking P. Foggia, C. Sansone, M. Vento.² Данная библиотека была разработана авторами VF-алгоритма специально для тестирования производительности алгоритмов проверки изоморфизма графов. Графы в библиотеке различаются по структуре и размерам. Для каждого типа графов определенной структуры и размера в библиотеке содержатся по сто пар изоморфных графов.

Структура библиотеки

Библиотека состоит из следующих типов графов:

- Фиксированой степени (степени 3, 6, 9)
- Фиксированой степени с возмущениями
- k -мерные решетки (двумерные, трехмерные, четырехмерные)
- k -мерные решетки с возмущениями
- Случайные графы

1.1.1 Графы фиксированой степени

Графы фиксированой степени — это графы, в которых число ребер (входящих и исходящих) строго равно заданному числу. Такое число называется валентностью или степенью графа. В библиотеке содержатся графы фиксированных валентностей 3, 6 и 9, обозначаются b03, b06 и b09 соответственно. Данные графы получены вставкой ребер между произвольных вершин до тех пор, пока валентность каждой вершины не стала равна заданному числу.

1.1.2 Графы фиксированой степени с возмущениями

Для внесения некоторой нерегулярности были введены графы фиксированой степени с возмущениями. В таких графах средняя степень вершин ограничена, но валентность каждой вершины не ограничена. Для получения графа такого типа вначале был создан

¹<http://math.tsu.ru/mwc/>

²<http://amalfi.dis.unina.it/graph/db>

граф фиксированной степени. Далее некоторое число ребер было перемещено от своих вершин к другим вершинам. Число перемещенных ребер равно $M = 0.1 \cdot N \cdot V$, где N — число вершин, V — валентность. Следовательно перемещается 10% ребер исходного графа.

1.1.3 k -мерные решетки

Графы регулярной структуры являются худшим случаем для общих алгоритмов проверки изоморфизмов. Для таких типов графов обычно разрабатываются специальные алгоритмы. В библиотеке представлены дву-, трех- и четырехмерные решетки (обозначаются m2D, m3D, m4D). В двумерных решетках каждая из вершин соединена с 4 соседними вершинами (кроме вершин на границе графа), в m3D — с 6 вершинами, в m4D — с 8 вершинами.

1.1.4 k -мерные решетки с возмущениями

Решетки с возмущениями получаются добавлением в исходную решетку равномерно распределенных ребер. Число ребер для добавления — $\rho \cdot N$, где N — число вершин, ρ — коэффициент, больше 0. Чем больше ρ , тем больше нерегулярности в графе. В библиотеке представлены графы со значениями $\rho = 0,2$, $\rho = 0,4$, $\rho = 0,6$ (r2, r4 и r6 в названии графа).

1.1.5 Случайные графы

Случайными графами называются графы, в которых вершины соединены без определенной структуры. При создании таких графов задается число η , которое является вероятностью нахождения ребра между двумя вершинами n_1 и n_2 . Число ребер в графе — $\eta \cdot N \cdot (N - 1)$, где N — число вершин графа. Если полученный граф не является связным, то в него добавляются ребра, чтобы сделать его связным. В библиотеке представлены графы со значениями $\eta = 0,01$, $\eta = 0,05$ и $\eta = 0,1$, обозначаются r001, r005 и r01 соответственно.

1.1.6 Обозначение графов

Название серии графов строится таким образом:

$$\langle T \rangle \langle P \rangle _ \langle N \rangle$$

Где:

T тип графа;

P опциональный дополнительный параметр;

N число вершин.

Пример: b09m_60 — серия графов ограниченной степени 9, с возмущениями, 60 вершин.

1.2 Запускаемые алгоритмы

При проведении вычислительного эксперимента запускались следующие алгоритмы:

- VF
- Рассматриваемый в статье (VF с матричным фильтром)

VF

VF³ алгоритм является backtracking алгоритмом поиска изоморфизма графов.

VF с матричным фильтром

Рассмотренный в статье алгоритм является модификацией VF алгоритма. Была использована функция выполнимости (feasibility function в исходной статье), основанная на идее прямого алгоритма поиска изоморфизма графов. В основе функции лежит решение системы линейных уравнений методом Гаусса-Зейделя. Решение СЛУ можно довольно эффективно распараллелить, с другой стороны схему backtracking нельзя достаточно эффективно распараллелить из-за того, что на каждой итерации алгоритма неизвестно, сколько вершин для последующей работы будет найдено. Работа функции очень сильно зависит от точности решения СЛУ. Если решение недостаточно точное, то будут выбираться неподходящие вершины, а подходящие наоборот, не будут выбраны. В вычислительном эксперименте точность была выбрана $eps = 10^{-10}$. На всех сериях графов алгоритм запускался на 20 процессорных ядрах.

Распараллеливаемость алгоритмов

Основанные на схеме backtracking алгоритмы не могут быть эффективно распараллелены из-за особенностей схемы: дерево поиска не может быть построено полностью, на каждой итерации алгоритма появляются новые ветвления. В нашем алгоритме распараллеливается не backtracking схема, а другая очень затратная операция — выбор подходящих вершин на добавление в дерево поиска. В процессе выбора вершин решается СЛУ методом Гаусса-Зейделя, данный алгоритм достаточно хорошо распараллеливается.

1.3 Результаты вычислительного эксперимента

В ходе проведения вычислительного эксперимента было обчислено на изоморфизм серии графов из библиотеки VF. Учитывалось время установления изоморфизма и число возвратов по дереву поиска. Результаты вычислительного эксперимента представлены в Табл. 1. Как видно из таблицы, в среднем число возвратов при использовании матричного фильтра меньше, чем у стандартного VF алгоритма, а на некоторых сериях равно нулю.

³<http://amalfi.dis.unina.it/graph/db/papers/vf-algorithm.pdf>

Серия	VF		McKay		Матричный фильтр	
	время	число	время	число	время	число
b03_20	0,04276ms	4,52			3,46ms	2,7755
b03_60	0,1049ms	16,47			87,6ms	10,0408
b03_100	0,1840ms	26,21			655,6ms	16,03
b09m_20	0,050616ms	0,101			5,7ms	0
b09m_60	0,139ms	0,356			241,3ms	0
b09m_100	0,2563ms	0,33			2453ms	0
m2Dr2_16	0,03613ms	1,88			2,66ms	1,175
m2Dr2_64	0,10743ms	13,75			153,4ms	5,91
m2Dr2_100	0,19735ms	32,31			669,7ms	16,95
m3D_27	0,04334ms	0,36			7,06ms	4,38
m3D_64	0,1013ms	4,3			222,29ms	33,80
m3D_125	0,28176ms	34,93			11650ms	414,41
m3Dr6_27	0,0503ms	1,14			8,58ms	0,289
m3Dr6_64	0,11702ms	2,73			205,9ms	0,454
m3Dr6_125	0,27926ms	6,09			3726ms	1,91
m4Dr4_16	0,0387ms	1,51			2,82ms	0,97
m4Dr4_81	0,14986ms	3,16			448,02ms	3,97
r001_20	0,03743ms	1,65			3,19ms	0,968
r001_60	0,09475ms	10,98			11195ms	3858
r001_100	0,170ms	7,162			21985ms	1362
r005_20	0,0362ms	0,7			3,50ms	0,776
r005_60	0,11336ms	0,63			132,8ms	0,0306
r005_100	0,283ms	0,45			2867ms	0
r01_20	0,0396ms	0,242			4,17ms	0,1
r01_60	0,1647ms	0,16			315,1ms	0
r01_100	0,422ms	0,16			4649ms	0

Табл. 1